REMARKS

Applicant respectfully requests reconsideration of this application. Claims 1-14, 16-19, and 21-30 are pending. Claims 2, 4, 6-8, 10-14, 16-19, 21-27, 29, and 30 have been amended. No claims have been cancelled. No claims have been added. Therefore, claims 1-14, 16-19, and 21-30 are now presented for examination.

Request for Continued Examination

Because of the expanded arguments provided here and the clarifications made to the claims, Applicant hereby request continued examination to allow for a full examination of the application and the claims.

Claim Rejection under 35 U.S.C. §103

Aoshima, et al. in view of Young

The Examiner rejected claims 1, 2, 4-8, 12, 19, and 21-27 under 35 U.S.C. 103(a) as being unpatentable over U.S Patent No. 5,510,859 of Aoshima, et al. ("Aoshima") in view of U.S Patent No. 6,560,606 of Young ("Young").

It is again respectfully submitted that Aoshima and Young do not teach or suggest the elements of the claims. The arguments previously presented in the prior response are clarified and expanded here.

Claim 1 reads as follows:

1. A method, comprising:
   receiving requirements for a plurality of modules;
   determining an inter-module dependency tree, the inter-module dependency tree being based on the requirements; and

modifying a module function in accordance with the inter-module dependency tree.

Claim 1 thus provides for "receiving requirements" for modules, determining a inter-module dependency tree "based on the requirements", and modifying a module function "in accordance with the inter-module dependency tree". When the cited references are closely examined, it is respectfully submitted that it can be seen that NONE of these elements are taught or suggested by the references.

Aoshima – Aoshima describes a program debugging support method and apparatus. For this reason, it was previously stated that Aoshima is not concerned with the dynamic operation of a program, but rather with expressions of the program for use in identifying errors. In response to the previous arguments, the Examiner stated that "the dynamic operation of a program" is not in the claims. It is respectively submitted that the Examiner's comments do not address the point of the argument. The Examiner has taken the words of the argument and interpreted them as if they were intended to be part of the claims themselves. The words "dynamic operation of a program" are not contained in the claims. However, in light of the Examiner's comments, the Applicant hereby attempts to clarify the arguments presented regarding Aoshima.

Aoshima provides that, to achieve the stated objects regarding specifying an execution position or error occurring position of a program, "a module relation diagram is formed which indicates a structure of an overall program and a hierarchical tree structure of a calling relationship of each module." Aoshima goes on to state that "[t]his module is displayed together with a source program. In addition, an execution position of the program on a position where an error occurs is displayed on a corresponding position of

the source program displayed and the module relation diagram." (Aoshima, col. 2, lines 22-31) Aoshima further states that "[t]o display the execution position on the module relation diagram, information indicative of the module calling relationship is stored beforehand in a tree table, according to which the module relation diagram is displayed." (Aoshima, col. 2, lines 32-36)

Therefore, Aoshima is a structure that shows the existing calling relationships for modules. This is used for use in creating displays and reports that are used for debugging and program analysis. This is further shown in the other portions of Aoshima that discuss the tree structure. For example, as provided in Figure 2 of Aoshima, when an error is found 511, a text table is formed 519 and displays regarding the execution of the program are provided 520-521. "In process 520, the error occurrence positions are displayed on both the module relation diagram and in the source text. Here, a reference is made to the previously set tree table and the contents of the text table." (Aoshima, col. 9, lines 62-65) The module relations and related tree structure shown in Aoshima are clearly intended to be tools for error checking and debugging of programs.

Thus, Aoshima fails to show any of the elements of the claims. With regard to receipt of requirements for a plurality of modules, the Examiner cites to various portions of the following:

> To achieve the above-described objects of the present invention, a module relation diagram is formed which indicates a structure of an overall program and a hierarchical tree structure of a calling relationship of each module. This module relation diagram is displayed together with a source program. In addition, an execution position of the program on a position where an error occurs is displayed on a corresponding position of the source program displayed and the module relation diagram.

> To display the execution position on the module relation diagram, information indicative of the module calling relationship is stored beforehand in a tree table, according to which the module relation diagram is displayed. With respect to each branch of the tree structure indicative of the calling relationship between modules, the tree table has an identification number for each branch of tree structure which shows the module calling relationship, a hierarchical level which shows a depth of calling, a calling sequence list of a current module and a display position of the module name of each branch with respect to each branch of the tree structure indicative of the module calling relationship.

(Aoshima, col. 2, lines 22-44) What this paragraph indicates is that module relation diagram is formed to indicate the structure of the program and a hierarchical tree structure for the calling relationship of modules. This provision contains no teaching or suggestion that requirements for modules are being received, or that a dependency tree is being formed based on such requirements. In Aoshima, the structure of a program already exists and the module relation diagram reflects this calling structure. As indicated in the quoted portion, information indicative of the module calling relationship is stored beforehand in a tree table, and according to this information the module relation diagram is displayed.

In this regard, it submitted that Aoshima demonstrates a different type of tree structure. Rather than being based on the inter-modules dependencies, the tree discussed in Aoshima illustrates the calling relationship between the modules. What the tree in Aoshima will show is how a program reached a point by showing which modules has called which other modules. While useful in debugging, this is not the same as providing an inter-module dependency tree. The use of a diagram to illustrate calling relationships does not provide for any of the elements of claim 1.

As was previously argued, Aoshima does not provide for modifying a module function in accordance with the inter-module tree. In response to this argument, the Examiner states that "Aoshima teaches a calling relationship of a function corresponding to the relation diagram (col 7, ln 62-65), the functions corresponding to the branches in the module [relation] diagram are executed a [plurality] of times." In Aoshima, a display illustrates a relation diagram (such as shown in the display 12 of Figure 1). Figure 7 then provides an example of a relation diagram 73. The cited portion of Aoshima simply indicates that a user program to be debugged is activated, and that a process includes extraction of a calling relationship of a function corresponding to the relation diagram. (Aoshima, col. 7, lines 62-65) Examples are also shown in Figures 8-10(B). Figure 16 then provides an example of tree data indicative of a module relation diagram. However, the examples provided do not teach or suggest the element of the claims. None provide for modifying a module function "in accordance with an inter-module dependency tree."

In a sense, the Examiner has a concept of the claims reversed. What Aoshima shows is how the described diagram reflects the calling of the functions. In addition to the other differences, what Aoshima does NOT show is any modification made in accordance with the diagram. The examples provided by Aoshima show no modifications of any kind. In short, Aoshima provides a diagram as a record of calling relationships. As indicated in Aoshima, what is presented is a "program testing and debugging support system" (Aoshima, col. 1, lines 7-8) and specifically providing "for retrieving an execution history after the program is executed" (Aoshima, col. 1, lines 11-13)

Young – It is submitted that Young does not teach or suggest the elements of the claims missing from Aoshima, as argued above. Young specifically relates to computer processing of metered information regarding communication services. Young contains no teaching or suggestion of an inter-module dependency tree based on module requirements, or modifying a module function in accordance with an inter-module dependency tree.

The Examiner has stated that Aoshima does not explicitly disclose that the diagram tree is a dependency tree, but that "Young teaches dependency (dependencies between there, col 3, ln 45-48)". However, the cited portion is not related to the concepts in Aoshima and does not provide any support for the rejection of the claims. What Young states is that "a mechanism is provided for tracking and enforcing the ordering of data processing by the processing modules so as to take into account dependencies between them." (Young, col. 3, lines 46-49) If the provisions of Young are in any way related to the concepts discussed here, it is simply to state that processing modules may have dependencies. This does add anything to Aoshima because, as shown above, Aoshima is illustrating a different type of tree that is formed under different circumstances and for a different purpose. The fact that dependencies might exist between modules cannot logically be used to make the calling relationship tree in Aoshima into a inter-module dependency tree.

For these and other reasons, Aoshima, and Young, separately or in any combination, do not teach or suggest the elements of the independent claims presented in the application. The rejected claims are dependent claims that are allowable as being dependent on the allowable base claims.

Further, it is again submitted that there is no motivation shown for combining Aoshima with Young. It is required that a motivation for combination of references be shown. The Office Action indicates that "the configuration storage would provide tracking and enforcing the ordering of data processing by the processing modules." Even if this is correct and relevant, this is not a showing of motivation. A showing that a combination provides an attractive result is not a showing of a motivation for the combination. Again, Aoshima describes a debugging process. Young regards processing data with multiple modules and counters, and specifically regards a metering and processing system for processing metered information that incorporates configurable processing modules and a configuration manager. There is no connection between these materials and they cannot be combined for purposes of obviousness.

It is submitted that the above arguments also applies to independent claims 12, 19, and 24 and such claims are thus also allowable. The remaining claims are dependent claims and are allowable as being dependent on the allowable base claims.

It is further submitted that the portions of the references cited to show many of the other claims appear to be unrelated to the claims. For example:

For claim 4, the Examiner has cited to a description of the identification numbering of a relation diagram to show association of a module command with an inter-module dependency. It s unclear how the numbering of the diagram would show such association. (Aoshima, col. 8, lines 17-25)

For claim 5, it appears that the Examiner has cited to display positions for function names in a tree table to show determining a phase for a command of a module.

It is unclear how the position of functions names in a tree structure relates to phases of commands. (Aoshima, col. 6, lines 56-68)

Claim Rejection under 35 U.S.C. §103

Aoshima, et al. in view of Young and APA

The Examiner rejected claims 3, 9-11, 13, 14, 16-18, and 28-30 under 35 U.S.C. 103(a) as being unpatentable over Aoshima in view of Young as applied to claim 1 and further in view of subject matter alleged to be admitted prior art (APA).

The subject matter described by the Examiner as the alleged APA does not teach or suggest the elements of the claims missing from Aoshima and Young, as argued above. For this reason, Aoshima, Young, and this material, separately or in combination, do not teach or suggest the elements of the independent claims presented in the application. The rejected claims are dependent claims that are allowable as being dependent on the allowable base claims.

Further, it is submitted that there is no motivation shown for combining Aoshima and Young with the alleged APA. It is required that a motivation for combination of references be shown. For example, the Office Action indicates that "because APA's the parameter may be preserved in some form of non-volatile storage would improve the efficiency of Aoshima's and APA's [Young's?] systems by initializing modules during device start-up." Even if this is correct and relevant, this is not a showing of motivation, but rather showing an advantage of the combination. A showing that a combination may provide an attractive result it is not a showing of a motivation for a combination. Again, Aoshima describes a debugging process. Young regards processing data with multiple modules and counters, and specifically regards a metering and processing system for

processing metered information that incorporates configurable processing modules and a configuration manager. The APA describes a background regarding module operations, including initialization, reconfiguration, and shutdown. There is no apparent connection between these materials and they cannot be combined for purposes of obviousness.

Conclusion

Applicant respectfully submits that the rejections have been overcome by the amendment and remark, and that the claims as amended are now in condition for allowance. Accordingly, Applicant respectfully requests the rejections be withdrawn and the claims as amended be allowed.

Invitation for a Telephone Interview

The Examiner is requested to call the undersigned at (303) 740-1980 if there remains any issue with allowance of the case.

Request for an Extension of Time

The Applicant respectfully petitions for an extension of time to respond to the outstanding Office Action pursuant to 37 C.F.R. § 1.136(a) should one be necessary. Please charge our Deposit Account No. 09-0457 to cover the necessary fee under 37 C.F.R. § 1.17 for such an extension.

Charge our Deposit Account

Please charge any shortage to our Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 10/12/05

Mark C. Van Ness
Reg. No. 39,865

12400 Wilshire Boulevard
7th Floor
Los Angeles, California 90025-1026
(303) 740-1980